Technical Report No. 49

COMPUTER PERSONNEL SELECTION AND CRITERION DEVELOPME III. THE BASIC PROGRAMMING KNOWLEDGE TEST

June 1966

ELECTRONICS PERSONNEL RESEARCH GROUP

Department of Psychology

UNIVERSITY OF SOUTHERN CALIFORNIA

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Technical Report No. 49

COMPUTER PERSONNEL SELECTION AND CRITERION DEVELOPMENT: III. THE BASIC PROGRAMMING KNOWLEDGE TEST

June 1966

Project Designation NR 153-093
Contract Nonr-228(22)

Principal Investigator
Joseph W. Rigney

Project Director
Raymond M. Berger

Research Associate
Robert C. Wilson

Computer Specialist
Frank Teplitzky

Prepared for

Personnel and Training Branch
Psychological Sciences Division
Office of Naval Research

Naval Command Systems Support Activity
Office of the Chief of Naval Operations

and

Psychological Research Branch
Bureau of Naval Personnel

DEPARTMENT OF PSYCHOLOGY

UNIVERSITY OF SOUTHERN CALIFORNIA

ACKNOWLEDGMENTS

We wish to express our appreciation to the many persons, representing computer user organizations, who contributed signficantly to this phase of the study; and to express our regret that space does not permit individual acknowledgment of each.

Mr. Jack Salvail of NAVCOSSACT, our project monitor, provided the assistance and encouragement that was indispensable to the success of the project.

Dr. Norman Willmorth of SDC, Mr. Guy Dobbs of SDC, and Mr. Charles Cree of IBM, our subject-matter experts, brought their high standards of quality to bear on the review and revision of many of the test questions.

More than 30 contributors translated the critical knowledge requirements in their field into test questions that formed the basis for a fair and representative programming knowledge test.

Captain H. S. Foote and Captain A. J. Henry, Commanding Officers of NAVCOSSACT during the course of this study, and Mr. Eric Wolf, the Technical Director of NAVCOSSACT, have our special appreciation for their parts in the initiation and active support of the project.

The management and programming personnel of both civilian and Naval organizations have our sincere thanks for cooperation in arranging or participating in the long hours of the preliminary or the final testing. A list of these organizations is given on the following page.

Mr. Richard McKillip of OIR directed the complex arrangements necessary for the testing of many Navy groups.

This research was greatly facilitated by the continued support of Dr. Glenn L. Bryan and Dr. James Regan, Personnel and Training Branch, Office of Naval Research; and Captain L. P. Wilder, Mr. Sidney Friedman, Dr. Victor Fields, and Dr. Martin Wiskoff, Psychological Research Branch, Bureau of Naval Personnel.

ORGANIZATIONS PARTICIPATING IN THIS PHASE OF STUDY

General

Applied Physics Laboratory

Bureau of the Census

Edgerton, Germeshausen & Grier (Nevada)

General Electric (Falls Church, Va.)

North American Aviation

Northrop

Planning Research Corporation (Washington, D.C.)

RAND

System Development Corporation

Navy*

Naval Command Systems Support Activity

Office of Industrial Relations

Bureau of Personnel

Bureau of Ships

Bureau of Supplies & Accounts

Bureau of Weapons

Bureau of Yards & Docks

Navy Finance Office

Marine Corps (Hdqtrs.)

---

*The designations listed are those existing at the time of participation in the study; most of the Bureaus are now designated as Commands.

ABSTRACT

This is a report on the criterion development phase of a long term research program concerned with computer personnel selection and evaluati

Two types of criterion measures are involved in this phase. Both are proficiency tests, one designed to test an individual's knowledge of the basic principles and techniques of programming, and the second, to test an individual's performance in depth in the systems analysis and systems design areas. The development of the first type, the Basic Programming Knowledge Test (BPKT), is described in this report. The second type of measure is now under construction and will be described in a subsequent report.

The BPKT is intended to stand by itself as a criterion of programming proficiency. To achieve a close correspondence of test content to programming job requirements, subject-matter experts participated in the construction and review of the test questions. Test questions were selected that met the criteria of discrimination and appropriate difficulty, as indicated by the statistical analysis of results of a large preliminary testing. The final form of the test consists of 100 multiple choice questions that are designed to be free of references to specific computers and languages now in use.

Normative scores have been developed for Navy computer groups. The relationships of the BPKT test scores to a number of vocational and educational variables are described. Recommendations are made for the use of the BPKT in personnel selection and evaluation of experienced programmers and analysts, and as a criterion measure against which aptitude tests may be validated.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF TABLES

LIST OF FIGURES

COMPUTER PERSONNEL SELECTION AND CRITERION DEVELOPMENT: III. THE BASIC PROGRAMMING KNOWLEDGE TEST

SECTION I. INTRODUCTION

This is the third report of a long-term research program devoted to computer personnel selection and evaluation. The program is divided into three phases: 1) job analysis - to describe what programmers and analysts do; 2) criterion development - to measure the effectiveness of experienced programmers and analysts; and 3) aptitude test development - to select inexperienced individuals for training.

The job analysis phase was described in two previous technical reports (Rigney, Berger, Gershon, 1963; Rigney, Berger, Gershon, Wilson, 1963). This report is concerned with the criterion development phase, specifically the building of a proficiency test, the Basic Programming Knowledge Test (BPKT), as one type of criterion measure of programming performance.

There are various kinds of criteria in use for evaluating computer personnel, but, as in other technical fields, most of these criteria are not entirely satisfactory. Criteria that depend on subjective evaluation by supervisors, on salary and experience levels, on job classification, or on productivity and error rate measures, are either inappropriate, unreliable, or lack standardization from company to company or even from section to section within a company. It is proposed that a way out of this dilemma is to use a proficiency test as a criterion measure. This report describes the development of such a

test, involving a comprehensive job analysis, statistical analysis of test scores, and review by the intended user at all stages.

The BPKT was planned to meet three needs. The first was for an instrument to use for selecting experienced personnel; the second was for a method to assist in classifying, evaluating, and upgrading programmers and analysts; and the third was for an objective, reliable research instrument to be used in validating aptitude tests or other predictors.

The management of NAVCOSSACT[1] expressed the need for an objective method to aid in the selection and in the evaluation and classification of applicants for programmer and analyst positions. The computer field has drawn to it a vast assortment of individuals of varying education and experience. There is no standard computer training curriculum, and many of the people doing programming or analysis learned the skills partially or completely on the job. In addition, people vary in their ability to acquire programming or analyst skills, and in their ability to transfer what skills they have acquired from one situation to another. NAVCOSSACT supervisors needed to supplement subjective evaluation of an applicant with some objective assessment of his programming proficiency, particularly those skills most useful in NAVCOSSACT work. The BPKT was constructed to measure those skills.

---

[1]This is the Naval Command Systems Support Activity. NAVCOSSACT is a computer-centered activity that supports the Chief of Naval Operations program of establishing and maintaining an Automatic Data Processing System Network for Naval command and control systems. The systems analysis and programming efforts in the various ADP projects are carried out by coordinated teams of military, civil service, and contractor personnel at NAVCOSSACT.

The BPKT measures proficiency primarily at the lower and intermediate programming levels. Plans for a second type of proficiency test, specific to the areas of systems analysis and systems design are discussed in Section V of this report.

The research use of the BPKT as a criterion against which aptitude tests and other predictors may be validated will be an important part of the overall research program. A battery of aptitude tests has been systematically administered to starting classes of trainees in a Navy programming course. The tests will be correlated later against results with the BPKT given to the same personnel when they are on the job. Aptitude tests were also administered to participants in the norming of the BPKT to collect concurrent validity information. The results of this latter testing are discussed in Section III.

SECTION II. TEST DESIGN AND DEVELOPMENT

The Basic Programming Knowledge Test was designed to sample a broad range of programming information, techniques, and applications. Fundamental to the design purpose was the representation of the most commonly exhibited programming behavior and knowledge. The construction of each test question started with a task statement or set of task statements that had been shown to be descriptive of a part of most programming jobs. Subsequent development of the test questions involved a series of checks on their suitability for eliciting and measuring the described programming behavior or knowledge. Since the validity of the BPKT depends in large measure on sampling the subject matter that underlies job performance, the procedure for achieving this content validity is of importance.

Test Item Specifications

The content of the BPKT was based on those programming activities and tasks which a large group of programmers said they spend a moderate or great amourt of time doing. It was assumed that if a large proportion of programmers say they do a particular task, then the knowledge and skill required for performing the task is basic to the programming field.

With the assistance of experts experienced in computer work an effort was made to develop an exhaustive list of statements describing the tasks which computer personnel perform. Each statement begins with

a verb, such as "develop" or "code" or "design" and consists of one or two brief sentences. After several revisions and tryouts, a final list of 186 task statements was developed as the basic tool for job description and classification.[2] Following are examples of task statements:

Code in compiler language.

Make corrections to own or other's programs by reassembly or recompilation of corrected source-language code.

Develop program production schedules.

Assign people with appropriate skills and talents to tasks that require these skills.

A pool of the most often selected task statements was established by using job analysis data. The procedure for making up the pool consisted of finding the task statements which were rated by fifty per cent or more of the programmer group as activities they spent a moderate or large amount of time doing. This procedure was first carried out with the 50 programmers from sixteen organizations that participated in the job analysis phase of the research. The same procedure was then applied to a second group made up of 21 programmers from NAVCOSSACT.

There was a considerable overlap of the tasks selected by both groups, indicating the general applicability of a test based on these tasks to programming personnel in groups outside of NAVCOSSACT. Since one of the primary intended uses of the BPKT was for screening programming applicants to NAVCOSSACT whose experience was obtained elsewhere,

---

[2] The entire list of 186 task statements is given in Technical Report No. 36, and the results of the job analysis are presented in Technical Report No. 37.

the underlying communality of activities tested would make the test fair as well as effective as a selection instrument. The pool consisted of 59 task statements that were common to programming jobs in as well as outside NAVCOSSACT.

Categories of Programming Tasks

This pool of task statements was divided into six general categories. These categories provided convenient groupings of tasks for test question construction. The six categories of tasks are described below:[3]

1. Logic, Estimation and Analysis. These tasks involve analyzing a problem into its component parts and converting them into a logical and efficient sequence of operations, and selecting the most appropriate method when considering computing costs, programming ease, or the user's requirements.

2. Flow Diagramming. These tasks deal with interpreting and constructing both general and detailed functional flow diagrams and program flow diagrams.

3. Programming Constraints. These tasks are concerned with the constraints imposed on programming by computer characteristics such as storage allocation, computing speed, and data format.

4. Coding Operations. These tasks deal with numeric machine code, assembly, and higher-level languages. Knowledge of various coding techniques and their relative merits is involved.

5. Program Testing and Checking. The tasks in this category involve test planning, programmed checks, debugging by use of diagnostic techniques such as dumps, traces, and snapshots, and code

[3]The list of task statements in each category is presented in Appendix A.

corrections by selecting the appropriate octal and symbolic sequences of code to correct a program error.

6. Documentation. These tasks involve the preparation of various types of documentation for different types of audiences, such as management, users, operators, designers, and program maintenance personnel.

Test Item Format and Generality

The BPKT is intended primarily as a test of breadth of knowledge of the commonly used principles and techniques of programming. The multiple-choice format lends itself well to making this kind of assessment. Performance as well as knowledge is measured to some degree by the multiple-choice questions in the BPKT. The solutions to many of the problems require programming performance behavior (for example, tracing through a flow diagram).

Perhaps the most difficult part of constructing and revising the test questions was to keep them free of references to any particular computer or programming language currently in use. Since it is doubtful that questions relating to programming ever can be divorced completely from computer characteristics and language logic, an attempt was made to let only those computer characteristics that are general to all computers, or to the most commonly used computers, influence the test questions. The same was done with respect to type of programming language.

Test questions were tried out on programming personnel from many different organizations with a variety of computer types and programming languages represented. During these try-outs, the examinees were asked to comment on any test questions they felt required special knowledge

connected with a particular computer or language. These comments, along with item-analysis information derived from each try-out group, were used to eliminate or to revise the biased questions. The test questions were further screened for biasing influences by two independent panels of computer experts, and by a preliminary testing of the BPKT with a relatively large number of programmers and analysts from several different organizations.

Certain kinds of programming tasks could not be tested without specifying characteristics of a computer and language. The device of a "hypothetical computer" was employed to allow everyone to start out on the same footing in attempting the problems, and to provide a specific computer-context for the questions. A separate section of the BPKT was designed in which the questions were organized around sequences of hypothetical mnemonic code for a hypothetical, binary computer.

Test Item Selection

Development of the Test Item Pool

Approximately 500 test questions were contributed by about 30 programmers and analysts. Most of the contributors were from NAVCOSSACT. These questions were reviewed and edited by a group of three programming consultants. The 250 questions which survived the review procedure were then pre-tested in sets of approximately 50 questions, with small groups of programmers. The groups, varying in size from 14 to 40, included programmers attending computer conferences or working in computer facilities on the West Coast.

Different sets of questions were administered to these groups and the results were analyzed to discover those questions that were too easy or that failed to discriminate in the right direction between high scoring and low scoring individuals. Many test questions were revised on the basis of the item-analysis results and the examinees' comments and were pretested further.

The results of the pretesting were reviewed by a committee of 12 programmers and analysts at NAVCOSSACT. The committee recommended 183 of the items as likely candidates for inclusion in the final test form. The final form was planned to consist of 100 items.

Preliminary Testing and Analysis

To reduce the 183 item pool to 100 items, another preliminary testing was carried out. This time to get greater stability and reliability of the results, a relatively large sample of computer personnel was tested on all 183 items. There were 120 participants drawn from eight organizations. Fifty-one were classified as programmers and 69 as senior programmers and analysts. The programming experience ranged from 4 months to 9 years with a median of 3½ years. Business, scientific, and military systems programming areas were each represented by about one third of the group.

The results of this last preliminary testing were analyzed by obtaining difficulty and discrimination indices for each item. These indices were used as the primary criteria for retaining or eliminating test questions. A test question was eliminated as too easy if more than 80 per cent of the group answered it correctly, and as too difficult if less than 30 per cent answered it correctly.

Each test question that met the difficulty criterion by being in the 30 per cent to 80 per cent range was then inspected to see if it met the discrimination criterion. The sample of 120 people was divided equally into upper and lower groups on the basis of the total scores made. Questions were selected for the final form of the test when the proportion of high scorers getting the question right was significantly larger than the proportion of low scorers getting the question right. The statistical significance of these differences was determined by computing phi coefficients (Guilford, 1965).

Description of the Basic Programming Knowledge Test

The Basic Programming Knowledge Test consists of 100 multiple-choice questions, divided into two sections. Section I contains 75 questions that range over many kinds of programming activities. The content has been generalized, so that knowledge of a particular computer or programming language is not required to answer the questions. Section II consists of 25 questions oriented around a hypothetical computer and a given set of computer instructions. A summary of the characteristics of the hypothetical computer and a list of the computer instructions are provided the examinee in a separate booklet that may be referred to in working out the solutions to the problems. Coding and debugging activities are emphasized in this second section. The examinee is required to work out each solution completely and accurately to arrive at the correct answer.

The time allowed for the 75 items in Section I is 75 minutes. The time limit for the 25 items in Section II is 60 minutes, including the time taken to look over the summary of computer characteristics and the instruction list. Since the items are not arranged in order of

difficulty, and some examinees tend to spend more time than they should on the difficult ones, the time left to finish the section is announced at given intervals.

The total administration time for the two sections is usually 2½ hours, including 2¼ hours of actual testing time, 10 minutes for distributing the test booklets and reading of the instruction page on the front of the booklet, and a five minute break between the two sections of the test. The two sections were designed to be independent of each other, separately scored, and separately timed. Either section may be given first, and separate testing sessions may be scheduled for the two sections, such as the morning and afternoon of the same day, or two mornings on succeeding days. Depending on the use of the test, the group to be tested and the time limitations, one section or the other may be dropped.

Internal-consistency reliability estimates[4] for the two sections of the BPKT were computed from the test results with the two norming samples described below. Section I reliability estimates for the NAVCOSSACT sample and the general Navy sample were .86 and .90, respectively. The Section II reliability estimates for the same samples were .85 and .68. The latter reliability value is probably an underestimation due to the item difficulty of Section II for the general Navy sample.

The correlation of Section I and Section II was .49 for NAVCOSSACT and .51 for the general Navy sample. This indicates that 25 per cent

---

[4]The internal-consistency reliability estimates were based on Kuder-Richardson formula 20.

of what the two sections measure is related (the correlation coefficient squared). For the most part the two sections are measuring different programming knowledge and skills.

Norming

Norming is a procedure in which a scale is developed to provide a meaningful basis for interpreting test scores. It makes it possible to determine the relative standing of an individual within his reference group. Test scores collected for the group are established along various points of a scale from highest to lowest, and any individual's score then can be compared directly with these scores. A percentile scale was used in the norming procedures for the BPKT. Thus, a percentile of 81 would mean the examinee made a score better than the scores of 81 per cent of his group. Since an individual's percentile score could be high with one group and lower with another, it is important that the norms for the appropriate group be used.

Norms for the final form of the test were developed on two groups. The first was a sample of 95 civilian programmers and systems analysts at NAVCOSSACT, and the second was a sample of 366 civilian programmers and systems analysts drawn from seven Navy bureaus at eight geographical locations.

A brief description of the norming conditions and the samples is given below. The actual normative data, including the centile ranks, means, median, standard deviation and score range for each sample are given in a data supplement to this report. The data supplement will have a restricted distribution.

Norming of a NAVCOSSACT Sample

Arrangements were made at NAVCOSSACT to test as many programmers and analysts as possible over a two-day period. Each testing session lasted a half day and two groups of 10 to 30 people were tested in each session. In addition to the BPKT, several aptitude tests were administered to collect concurrent validity data for these tests. The complete administration time for the BPKT, the aptitude tests and the vocational and educational data form was four hours. The testing conditions were excellent, and on the whole the motivation of the individuals during the test appeared to be good.

Well over half of the NAVCOSSACT personnel actively engaged in computer programming or analysis were able to participate in the testing. There were 129 people in all, 34 military and 95 civil service personnel. Contractor personnel were not included in this testing. Only the civilian personnel were included in the norm sample.

Norming of a General Navy Sample

Computer groups in Navy installations outside of NAVCOSSACT constituted the other major norm sample. In collaboration with the Office of Industrial Relations of the Navy, the BPKT was administered to 458 programmers and analysts from several Navy Commands[5], and from the Marine Corps. The examinees were located in headquarters activities in Washington, D.C. and in field activities in Philadelphia, Norfolk, Oakland, San Diego, Point Mugu, and Port Hueneme.

---

[5]The Navy Bureau designations at the time of testing were Ships, Yards and Docks, Supplies and Accounts, Weapons, Personnel, and Finance.

Not all of the people who participated in this testing were included in the development of the norms. Eliminated from the norms sample were the scores of 40 individuals with less than 1 year of programming experience or whose jobs were almost entirely managerial, and 52 military personnel. The last named group was not sufficiently represented in the testing, since the arrangements were directed primarily toward the Navy civilian employees.

The BPKT scores for the general Navy sample have been related to other data collected from each individual at the time of testing. A discussion of the relationships with measures of experience, training and education, and aptitude tests is given in the next section.

SECTION III. RELATIONSHIP OF THE BPKT TO EXPERIENCE, TRAINING, AND APTITUDES

Data are presented in this section showing the relationships between scores on the test and the experience, training and aptitudes of persons who have taken the test.[6] The data discussed are for the general Navy sample.[7] These data provide indirect evidence of the content validity of the test. If the test measures breadth of programming knowledge, then examinees with broader experience should score higher on the test than those with narrower experience. If the test measures level of programming skill, then those examinees who have experience in jobs requiring a high level of programming skill should score higher than those who were in jobs requiring a low level of programming skill.

Results of the analysis are presented under the following headings: Type of Experience, Level of Experience, Amount of Experience, Type of Training, Education and Aptitudes. For ease of interpretation, the results are given in the tables as percentages of examinees who scored above the median of the total group. The significance of differences was evaluated by (two-tailed) chi-square median tests (Guilford, 1965). The results for Section I of the BPKT are presented first.

---

[6] A copy of the Educational and Vocational Data form is given in Appendix B.

[7] Data for the NAVCOSSACT sample are presented in a data supplement.

Type of Experience

Type of experience was noted by asking the examinees to list the computers they used, the programming languages they used, their area of programming, and their primary job activity.

Computer Used

Performance on Section I of the BPKT was related to the type of computer with which the individual worked, binary or decimal (character). The percentage of examinees with experience on binary computers who scored above the median on Section I was signficantly larger than the percentage of those with experience on decimal computers only (Table 1, Sixty-four per cent of the 155 examinees whose work involved binary or both binary and decimal machines scored above the overall median. Only 40 per cent of the 211 examinees who worked solely with decimal machines scored above the overall median. The probability that a difference this large would occur by chance is less than 1 in 1000 ($p < .001$).

Programming Languages Used

Examinees were categorized as to whether they used problem-oriented languages, symbolic and numeric languages, or both. Table 1 shows the percentage of each group that scored above the median.

There were significant differences among the groups. Examinees who indicated they used only a problem-oriented language or no programmin language were least likely to score above the median. Those who used symbolic or numeric languages were about equally likely to score high or low. Those who used both a problem-oriented language and a symbolic or numeric la[illegible]e were more likely to score high.

Table 1

BPKT Section I
Relationships with Type of Experience

Variable	N	% Above Median	Chi Square	p
Computer Used			19.74	<.001
Binary or Binary & Decimal	155	64		
Decimal (character) only	211	40		
Programming Language Used			20.93	<.001
No Language Indicated	32	20		
Problem Oriented Language (POL)	51	33		
Symbolic or Numeric	144	51		
POL & Symbolic or Numeric	139	61		
Area of Programming			12.32	<.01
Data Manipulation	301	47		
Utility Programming	35	57		
Math or Scientific	25	84		
Primary Job Activity			14.81	<.001
Systems Analysis & Design	93	40		
Lead-Programming & Supervising Program Production	51	73		
Detailed Program Design & Coding	159	47		

Area of Programming

An examinee's area of programming was significantly related to his score on Section I of the BPKT. Those who worked in mathematical or

scientific programming were much more likely to score high. Those who worked in utility programming were somewhat more likely to score high, and those who worked in data manipulation were about equally likely to score high or low.

Primary Job Activity

The examinees were asked to indicate the activity that described the largest part of their job. The three most often-indicated activities are shown in Table 1. Examinees whose major duties were systems analysis and design were more likely to score low; those whose major duties were lead programming or supervising program production activities were more likely to score high; and those whose major activities were detailed program design and coding were about equally likely to score high or low.

Some of those whose primary activities were systems analysis and design either had little programming experience or their programming experience was not recent. On the other hand, individuals whose primary activities were lead programming and supervising the production of programs were likely to have been more directly involved in programming activities requiring a high level of programming proficiency.

Level of Experience

Civil Service grade and job titles were used as indices of level of experience. It was found that higher civil service grade levels were somewhat associated with greater programming proficiency, but when job titles were taken into account, the relation between grade level and programming proficiency did not always hold. The general Navy sample was a heterogeneous group which included both programmers

and analysts. The programmer group was divided into programmers and senior programmers, and the analyst group was divided into those who had previous programming experience and those who did not. Many of these systems analysts were employed at high grade levels because of their superior knowledge of programming; many others were at high grade levels because of their knowledge of a particular subject-matter field rather than proficiency in programming.

Civil Service Grade

The examinees were categorized by Civil Service grade (GS 7 to 9, GS 11, and GS 12 and above). Those examinees in the higher grades were more likely to score high on the test than those in the lower grades ($p < .05$) as shown in Table 2.

Job Title

Examinees were also categorized according to the job titles they listed. Table 2 indicates that programmers in the lower grades were about equally likely to be high or low performers on Section I of the BPVT. Senior programmers (GS 11+) showed a slight tendency to score high. Systems analysts (GS 11+) who had previously held the title of programmer were signficantly more likely to score high. However, systems analysts (GS 11+) who had never been programmers were significantly more likely to score low.[8]

[8] Ten systems Analysts were at the GS 7-9 level and were not included in the job title analysis because of the small size of the group.

Table 2

BPKT Section I
Relationships with Level of Experience

Variable	N	% Above Median	Chi Square	p
Civil Service Grade			6.14	<.05
GS 7 - 9	84	42		
GS 11	166	49		
GS 12 - 14	116	58		
Job Title			17.66	<.001
Programmer (GS 7 - 9)	76	47		
Senior Programmer (GS 11+)	174	56		
Systems Analyst (GS 11+, Previous Programming Title)	52	65		
Systems Analyst (GS 11+, No Previous Programming Title)	54	28		

Amount of Experience

Amount of experience was found by asking each examinee to list all the programming or analyst job titles he had held (in his present organization and other organizations) and the number of months he had worked with each job title.

Length of Experience

Length of experience was determined by the total number of months the individual had spent in computer-related jobs. There was a slight

tendency for individuals with only one or two years experience to score below the median on the test.

The relationship between length of experience and performance on Section I of the BPKT is not a linear one. The complete frequency distribution of test scores plotted against the distribution on experience by years shows a tendency for the average score to increase from the first year to the third year of experience. Beyond the third year, the average score for each year tends to remain about the same, slightly above average.

Perhaps the reason that the relationship between length of experience and test performance was not more pronounced is that length of experience is not a direct index of level of skill or breadth of knowledge. With increased experience and seniority, many individuals move away from direct concern with programming to other types of activities. Others work in programming jobs which make few demands for learning new skills and techniques. Individuals in such jobs are in the situation of the five-year man who has had one year of experience five times.

Number of Job Titles

Table 3 shows a highly significant relationship between performance on Section I of the BPKT and the number of job titles the individual has held. The greater the number of job titles held the greater the likelihood of a high score. It should be noted that having held a large number of job titles did not necessarily represent job-hopping.

Table 3

BPKT Section I
Relationships with Amount of Experience

Variable	N	% Above Median	Chi Square	p
Computer-related Experience			4.63	<.05
1-2 years	93	41		
3 or more years	273	53		
Number of Computer-related Job Titles			27.77	<.001
1 Job	128	39		
2 Jobs	125	48		
3 Jobs	64	55		
4-6 Jobs	44	84		

In many instances it meant that the individual had moved up through several positions in a single organization.

Type of Training

The examinees were asked to indicate whether they had received training in General Programming Procedures, Systems Analysis, Numerical Analysis, Computer Operation, Machine Languages or Machine-independent Languages. Performance on Section I was unrelated to training in General Programming Procedures, Systems Analysis, Numerical Analysis, and Computer Operation. However, those who said they had received

Table 4

BPKT Section I
Relationships with Type of Training

Variable	N	% Above Median	Chi Square	p
General Programming Procedures			.68	n.s.[9]
Trained	341	51		
Untrained	25	44		
Systems Analysis			1.70	n.s.
Trained	232	53		
Untrained	134	46		
Numerical Analysis			3.72	n.s.
Trained	64	61		
Untrained	302	48		
Computer Operation			1.59	n.s.
Trained	197	53		
Untrained	169	47		
Machine Languages			34.16	$< .001$
Trained	287	58		
Untrained	79	22		
Machine-independent Languages			22.06	$< .001$
Trained	118	68		
Untrained	248	42		

[9] n.s. means that the difference is not significant ($p > .05$).

training in Machine Languages or in Machine-independent Languages were significantly more likely to score high than were those who did not. These results are summarized in Table 4.

Education

The relationships of performance on the BPKT, Section I to education variables are shown in Table 5. It is evident that the higher the level

Table 5

BPKT Section I
Relationships with Education

Variable	N	% Above Median	Chi Square	p
Level of Education			49.45	<.001
High School or less	154	29		
1-3 years of college	97	59		
Bachelor's Degree	86	70		
Post-Graduate Work	29	76		
College Major			6.54	<.05
Accounting and Business Admin.	62	39		
Social Science, Humanities & Educ.	50	50		
Math, Engr., Phys. & Biol. Science	58	62		

of education or the more science-oriented the college major, the more probable it is that an individual will score above the median.

Aptitudes

Several groups of examinees took selected aptitude tests in addition to the BPKT. The test names are listed below along with a brief description of the test content and the aptitude measured by the test.

TEST NAMES AND CONTENT	APTITUDES
[illegible] Ordering - List steps in appropriate [illegible] to complete a given project.	Logical Ordering
Symbol Grouping - Rearrange scrambled symbols in a specified order as efficiently as possible.	Symbol Manipulation
Circle Reasoning - Discover rules for making [illegible] in patterns.	Discovering Symbolic Patterns
Figure Matrix - Apply rules discovered from changes in rows and columns of a 3 x 3 matrix of figures to select a figure for a specified cell.	Figural Relations
Word Transformation - Indicate new divisions between letters in a series or words to make a new series of words.	Symbolic Redefinition
Vocabulary - Select the word that is most similar in meaning to a given word.	Verbal Comprehension

Four of the aptitude tests showed significant relationships to performance on the BPKT. That is, individuals who scored in the upper half on these aptitude tests were more likely to score in the upper half on the BPKT. As summarized in Table 6, the tests which showed significant relationships were Temporal Ordering, Symbol Grouping, Circle Reasoning and Figure Matrix. These results imply that performance on Section I of the BPKT is related to an individual's ability to reason logically and to manipulate abstract symbols.

The vocabulary test was administered to determine whether understanding directions or word meanings entered into performance on the BPKT.

The lack of a significant relationship indicates that differences in verbal comprehension ability are not an important factor in BPKT scores.

Table 6

BPKT Section I
Relationships with Aptitude Tests

Variable	N	% Above Median	Chi Square	p
Temporal Ordering			10.31	<.01
Lower Half	27	30		
Upper Half	29	72		
Symbol Grouping			16.14	<.001
Lower Half	28	25		
Upper Half	28	79		
Circle Reasoning			4.32	<.05
Lower Half	45	38		
Upper Half	48	58		
Figure Matrix			14.16	<.001
Lower Half	43	28		
Upper Half	49	67		
Word Transformation			0.18	n.s.
Lower Half	50	50		
Upper Half	44	48		
Vocabulary			2.69	n.s.
Lower Half	76	42		
Upper Half	73	55		

BPKT Section I Summary

If one were to make a summary sterectype of the characteristics of persons who are likely to score above average on Section I of the BPKT, one would say that such persons are more likely to:

1. work with a binary computer
2. use both a POL and a symbolic or numeric language
3. do mathematical or scientific programming
4. do lead programming or program production supervision
5. be a GS 12 or above
6. be a senior programmer or a senior systems analyst who has formerly been a programmer
7. have had at least 3 years experience
8. have held 3 or more job titles
9. have had training in machine languages and machine-independent languages
10. be a college graduate
11. have majored in mathematics, engineering, or science
12. score high on tests of logical reasoning and symbol manipulation abilities

The above stereotype should be interpreted with caution. Like most summaries, it does not do justice to individual cases, since it is based on differences between groups.

Relationships to Section II of the BPKT

Section II requires interpreting and debugging sequences of mnemonic code for a hypothetical computer. The relationships of experience, training and aptitudes to performance on this section of the test are the same as those for Section I with the following exceptions:

1. GS level is not related to scores on Section II.
2. Length of experience is not related.
3. Systems analysts with previous programming experience are equally likely to score high or low on Section II. They were more likely to score high on Section I.
4. Training in Computer Operation is related to scores on Section II.
5. Scores on the Vocabulary test are related to scores on Section II. The Vocabulary test measures verbal comprehension ability which is probably important in reading and understanding the instructions for the hypothetical computer.

SECTION IV. RECOMMENDATIONS FOR USE OF THE BPKT

It was the authors' intention to provide a criterion of programmer proficiency. The fulfillment of this intent rests on the assumption that because of the elaborate care taken in the translation of programming job requirements into test form, the test stands by itself as a criterion of programming proficiency. The acceptance of the BPKT as a criterion will make it a powerful instrument for the purposes of selection, evaluation, and placement of personnel. Recommendations with respect to the uses of the BPKT are given below.

Selection of Experienced Personnel

The BPKT is recommended for an important role in the selection of applicants for computer programmer positions. The test will be particularly useful as a screening device when there are a large number of qualified applicants for a position. The number of people to be interviewed could be reduced to those individuals who are above a given standard of technical proficiency as determined by a cut-off score.

For the selection of applicants for a higher level position, such as systems analyst, the BPKT could be used in combination with tests specially designed for that job area. Figure 1 presents a plan for using proficiency tests, singly or in combination, for selecting programmers and systems analysts. The selection of applicants for a programming position would involve the use of the BPKT only. Applicants for a system analysis position can be differentiated into two job categories: those whose job may require involvement in the programming portion of a project, and those whose job is primarily concerned with the subject-matter applications of a project.

Figure 1 outlines a sequential decision procedure. The BPKT would be the first stage test for those cases in which programming experience is a prerequisite. Candidates for system analysis positions who scored higher than the cut-off score would be given special-area tests (SAT) to obtain additional information to assist in the placement decision. The phrase "conditionally accept" in the figure means that other information, in addition to test scores, may be available to management. There are characteristics of applicants that cannot be revealed by tests; for example, physical appearance and skill in social interaction. To the extent that management emphasizes the importance of these characteristics this supplementary information will be important in determining the final decision about the applicant. These recommendations for the sequential application of criterion measures are somewhat analogous to sequential decision strategies discussed in Cronbach and Gleser (1965).

Evaluation of Personnel On-the-Job

The BPKT can be used to evaluate computer personnel presently employed who are being considered for reassignment, advancement, and training in special areas. Norms that are associated with the various job levels provide a basis for job classification.

In addition to using the test as an instrument of individual evaluation, it can also be used as one for company evaluation. That is, the general level of a company's computer personnel can be compared to that of organizations within the same area of computer application. The weaknesses and strengths of the company personnel might be explored through examination of the scores on particular sets of items. Recommendations for training would be a natural consequence of such an examinatio

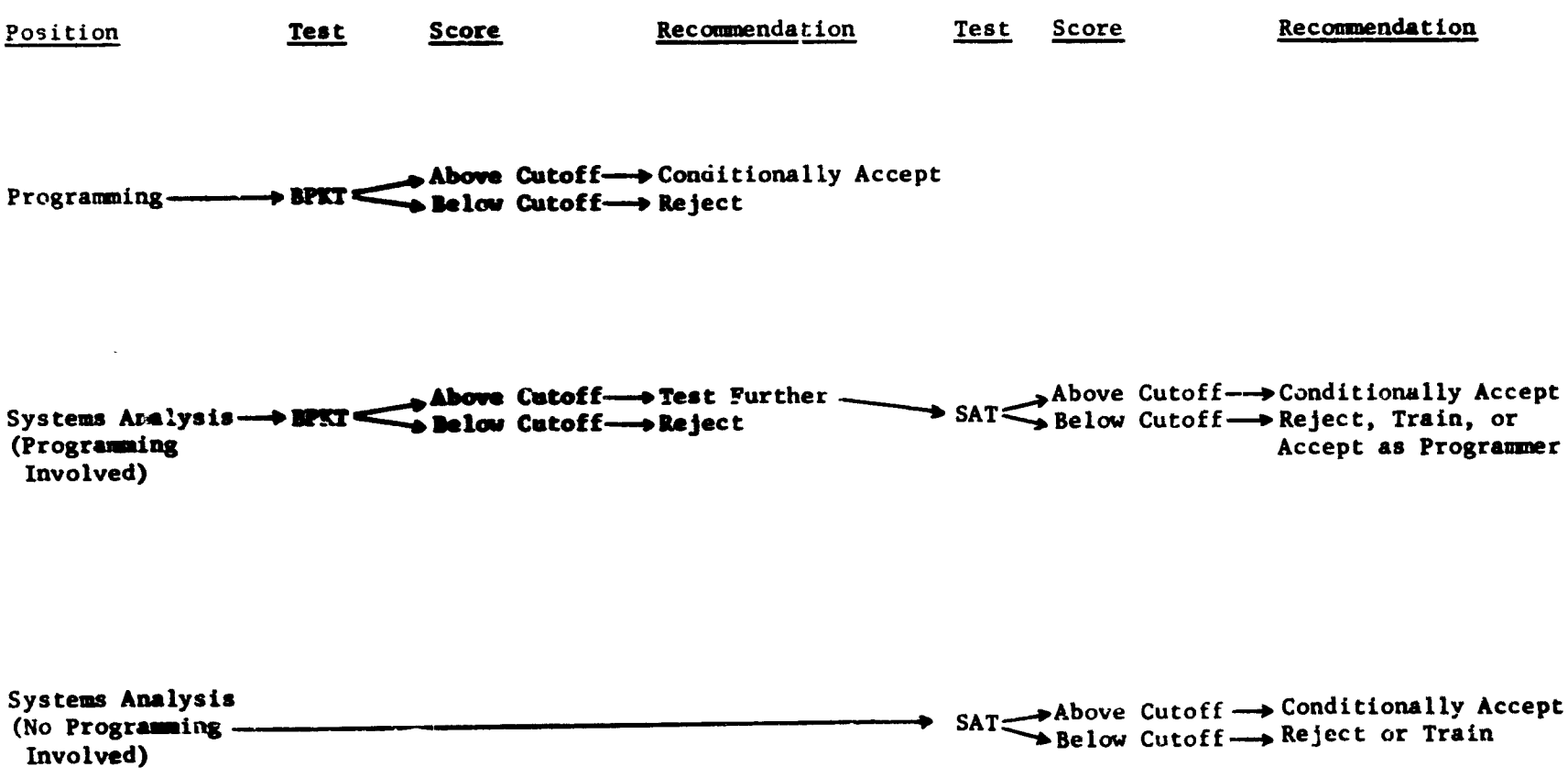


Fig. 1. Plan for selecting applicants for programming and systems analysis positions.

Certification of Programming Personnel

There is a need for setting up proficiency standards in the computer programming field. The BPKT could be useful as a certification instrument in the computer industry, just as proficiency tests are used in professional fields such as law and medicine.

Evaluation of Training Program Effectiveness

There is frequently a large disparity between an organization's training program and its actual job requirements. The BPKT could be used to make modifications in the training program by identifying those areas in which personnel on-the-job are inadequately prepared.

Validation of Aptitude Tests

For research purposes, the BPKT will serve as a criterion for determining how well aptitude tests can predict on-the-job success. Thus far, training grades have been the main basis for determining the degree of validity of aptitude tests. When ratings and other such measures have been used as criteria, the predictiveness of aptitude tests has been generally low. It is uncertain whether these results are due to the low reliability and inappropriateness of the criterion measures employed, or to the real lack of validity of the aptitude tests. The BPKT will serve as a reliable criterion for establishing such validity.

SECTION V. FUTURE PLANS

Maintaining, Updating, and Revising the BPKT

The BPKT as any other testing instrument must be maintained and updated for its most effective use. Maintenance of the present form requires that the base for NAVCOSSACT and other group norms be expanded for better interpretive use of the scores. When this data base is expanded, other information also should be collected, so that the BPKT can be correlated with other indices and predictors.

The updating of the BPKT is necessary to keep pace with the changing technology and skill requirements of the computer field. New test questions should be tried out during regular test administration and should replace those questions whose relevance has diminished.

A modification of the BPKT has been found necessary to make it more suitable for testing business-applications programmers. Thirty-five test questions in the BPKT will be replaced to make the test more appropriate for this area. The modified form should be ready for norming within the next year. The same maintenance and updating requirements would apply to this form, as well.

Special-Area Tests

The BPKT was designed to discriminate best at lower and intermediate levels of programming experience. It was recognized that no single criterion measure can be expected to discriminate equally well across the entire range of competence involved in programming and systems analysis. A second type of criterion measure was planned to provide

better discrimination at the higher job levels. This second type of criterion measure, covering the areas of systems analysis, systems design, and program design, is currently under development. Each test will probe in depth an individual's proficiency in a particular area of computer programming or analysis. These areas were defined by the job dimensions found in the job analysis. Consistent with the skill requirements of each job dimension, several typical problems are being constructed for each test. The special-area tests will require the individual to work out solutions to fewer but more complex problems than those included in the BPKT.

A sequence of six steps has been planned to complete the special-area tests: problem construction, problem review by a NAVCOSSACT committee, pretests of test problems, a final review by a NAVCOSSACT committee, collection of normative data on the sample previously tested with the BPKT, and analysis of the test data. A technical report will present results and recommendations for the use of this type of criterion measure in conjunction with the BPKT.

For research purposes, the systems analysis and systems design tests will serve as higher level criteria for validating aptitude tests and selection procedures. These special-area tests also will be used, in conjunction with the BPKT, in personnel selection and classification, personnel evaluation and upgrading, and training program evaluation.

REFERENCES

Cronbach, L. J. & Gleser, G. C. Psychological tests and personnel decisions. (2nd ed.) Urbana: University of Illinois Press, 1965.

Guilford, J. P. Fundamental statistics in psychology and education. (4th ed.) New York: McGraw-Hill, 1965.

Rigney, J. W., Berger, R. M., & Gershon, A. Computer personnel selection and criterion development: I. The research plans. Los Angeles: Univer. Southern California, Electronics Personnel Res. Group, February 1963. (Tech. Rep. 36) AD 299 244

Rigney, J. W., Berger, R. M., Gershon, A., & Wilson, R. C. Computer personnel selection and criterion development: II. Description and classification of computer programmer and analyst jobs. Los Angeles: Univer. Southern California, Electronics Personnel Res. Group, December 1963. (Tech. Rep. 37) AD 432 020

APPENDIX A

PROGRAMMER TASKS

Logic, Estimation, and Analysis

Task Statement Number	
47	Express solution to problem as sequence of logical steps.
48	Develop logical descriptions of functions to be programmed.
51	Specify the method to be used, from among the available methods, to perform a calculation or function.
55	Estimate running time and size of the computer program to be produced.
57	Estimate amount of computer time required for program check-out purposes.
76	Analyze anticipated program flow so as to utilize appropriate library subroutines as opposed to writing new subroutines.
77	Analyze frequency of use of subroutines to determine whether a "closed subroutine" or an "open subroutine" should be used and program accordingly.
78	From the routines available for a given method, determine which routines fit within the specific requirements of the problem, and pick one of these routines.
127	Study and analyze program description, flow charts, and listings as a prelude to program modifications or revisions.
130	Study and analyze program documentation as a prelude to the integration or utilization of the program in a program system.
131	Study and analyze programming language manuals (describing new or unfamiliar programming languages) in order to become proficient in the use of the language.

Coding Operations

Task Statement Number	
31	Translate detailed flow diagrams into machine language or symbolic language instructions.
32	Translate flow diagrams directly into machine language or symbolic language instructions without going through stage of detailed flow diagrams.
36	Code from program specifications in an intermediate symbolic language.
37	Code in symbolic assembly language for a simple one-to-one translation into machine code.
38	Code in a symbolic assembly language utilizing pseudo-instructions, macro-instructions, and systems routines.
39	Code in compiler language.

Documentation

Task Statement Number	
153	**Document program requirements.**
155	**Document program design specifications.**
157	**Document coding specifications.**
160	**Document finalised flow chart.**
161	**Document program description.**
164	**Document operating instructions.**
165	Document coding by including appropriate comments next to source language instructions.
166	Write program changes or change request document, e.g., error reports, revisions, modifications.

Programming Constraints

Task Statement Number	
79	Estimate minimum and maximum values in computations in order to avoid overflow stops, loss of significant digits, etc.
80	Inspect constants and computational sequences for possible overflow conditions.
81	Decide among alternative computational procedures for handling specific problems, e.g., multiple precision vs. single precision; round-off (or rounding) vs. truncation; floating vs. fixed, etc.
84	Analyze and plan programs so as to minimize required storage space and/or time (running time).
87	Decide among alternative formats for input or output information.
88	Design the program to meet the requirements of specified formats of input and output information.
91	Design and write program to insure that program size (number of registers) does not exceed space limit.
92	Allocate memory for a program within constraints of limited space and time.
94	List constants and parameters and specify their memory locations if necessary.
95	Break up large programs into self-contained pieces that can be accommodated by available memory.

Program Testing and Checking

Task Statement Number	
5	Devise program test plan.
6	Build consistency checks into a program. Examples: (1) to check whether input values of data to be processed are within the ranges expected by the programmer. (2) to check on the computed value of a payroll check to see whether it is less than zero.
7	Construct appropriate test data fully to check all paths of a program.
8	Locate "breakpoints" at most useful locations to obtain debugging information or data.
9	Code, in any language, special-purpose or "one-shot" testing routines for own program.
10	Debug own programs written in machine or assembly language, by "desk-checking."
11	Debug at machine language level own program written in automatic coding language.
12	Debug at source level own programs written in automatic coding language.
13	Debug at machine language level programs of other's written in an automatic coding language.
14	Debug at source language level programs written by others in automatic coding language.
15	Debug, by "desk-checking", programs of others written in machine or assembly language.
16	Spot check accuracy of own or other's programs away from the computer by checking code itself, using such cues as logical inconsistencies in the instructions, obviously incorrect addresses, etc.
18	Debug own programs by "stepping through" at console itself.
19	Spot check own or other's programs for accuracy, regardless of language level, by checking captured intermediate results for expected values.
40	Make corrections to own or other's programs by reassembly or recompilation of corrected source language code.
41	Make corrections to own or other's programs by numeric corrections of object program.
129	Study and analyze program documentation as a prelude to debugging or checkout of a program produced by someone else.

Flow Diagramming

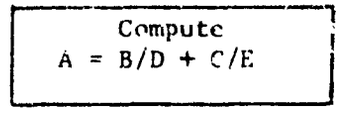
Task Statement Number

25 Given a functional description of program requirements generate functional block diagrams.

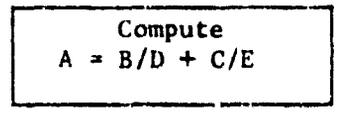
26 Translate functional block diagrams directly into machine instructions without going through flow and detailed flow diagramming stages.

27 Given program design specifications, generate program diagrams or flow charts.

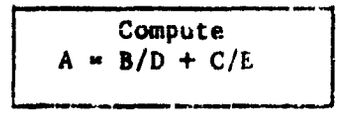
28 Translate broad system flow charts into flow diagrams with elements such as:



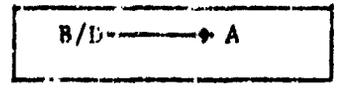
29 Translate functional block diagrams into flow diagrams, with elements such as:

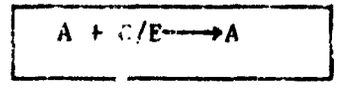


30 Translate flow diagrams with elements such as:



into detailed flow diagram elements, given in part below:





33 Generate a flow diagram from machine language or symbolic language instructions.

APPENDIX B

EDUCATIONAL AND VOCATIONAL DATA FORM

1. Test Booklet Number _____ Sex: M __ F __ Age _____

2. G.S. Level or Military Rank _____ Organization and Division _____

3. Highest School Grade (circle)
   High School: 9 10 11 12 College: 1 2 3 4 Post Grad: 1 2 3 4

4. Degrees

	Date	Major	Minor
None (year left college)	_____	_____	_____
Bachelor's	_____	_____	_____
Master's	_____	_____	_____
Doctoral	_____	_____	_____

5. Computer-related Training (check one or both columns that apply to your training.

	On-the-job training	Formal course
A. General programming procedures	( )	( )
B. System analysis	( )	( )
C. Numerical analysis	( )	( )
D. Computer operation	( )	( )
E. Machine languages	( )	( )
F. Machine-independent languages	( )	( )
G. Other (Specify) _____	( )	( )

6. No. of years since last formal classroom course _____

7. Current job title (give full title(s)) _____
   _____

8. No. of months with present job title (in this organization) _____
   Type of computer(s) involved in present position:
   _____

   Programming languages you use: _____

9. List previous programming/analytical job titles (in this and other organizations). Indicate also the approximate number of months in each and the types of computers worked with. Start with most recent and work back to first job in computer field.

Previous programming/analytic job titles	No. of Months	Types of computer(s) involved
_____	_____	_____
_____	_____	_____

10. Total no. of months in computer-related jobs (#8 and #9 combined): __

11. Which of the following are major parts or duties of your current job? Check column 1 for each item that applies. Of those checked in column 1, check column 2 for the ONE item that describes the largest part of your job.

	1	2
Program System Analysis	( )	( )
Program System Design	( )	( )
Program System Integration	( )	( )
(Integrating system components)	( )	( )
Planning and Scheduling Program Production	( )	( )
Supervising Program Production	( )	( )
Lead Programming Responsibility	( )	( )
Detailed Program Design and Coding	( )	( )
Debugging (Desk-checking)	( )	( )
Program Testing (Checking-out individual	( )	( )
programs or sub-systems)	( )	( )
Program System Testing	( )	( )
Program Documentation	( )	( )
Program Installation (Turnover)	( )	( )
Training (of other personnel)	( )	( )
Other (specify) _____	( )	( )

12. A. Which of the following programming areas have you been involved with to a significant degree in your current job? Check column 1 for each area that applies. Of those checked in column 1, check column 2 for the ONE that comprises the major area of your work

	1	2
Utility Program Development (General Purpose and Library)	( )	( )
Utility Program Development (Executive and Compiler)	( )	( )
Operational/functional Program Development (Math or Scientific)	( )	( )
Operational/functional Program Development (Data Manipulation, e.g., inventory control, Information Retrieva	( )	( )

B. Check below if you have been involved in the types of programming indicated in your current job.

Programming Real-time Systems	( )
Programming for Special Purpose Computers	( )

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

ORIGINATING ACTIVITY (Corporate author)	2a REPORT SECURITY CLASSIFICATION
Electronics Personnel Research Group University of Southern California Los Angeles, California	Unclassified 2b GROUP

REPORT TITLE

COMPUTER PERSONNEL SELECTION AND CRITERION DEVELOPMENT:
III. THE BASIC PROGRAMMING KNOWLEDGE TEST

DESCRIPTIVE NOTES (Type of report and inclusive dates)

Technical Report June 1966

AUTHOR(S) (Last name, first name, initial)

Rigney, Joseph W., Berger, Raymond M., Wilson, Robert C., and Teplitzky, Frank

REPORT DATE	7a TOTAL NO OF PAGES	7b NO OF REF
June 1966	42	4
8 CONTRACT OR GRANT NO Nonr-228(22) b PROJECT NO Same c d	9a ORIGINATOR'S REPORT NUMBER(S) Technical Report 49 9b OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10 AVAILABILITY/LIMITATION NOTICES

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

11 SUPPLEMENTARY NOTES	12 SPONSORING MILITARY ACTIVITY
	Personnel and Training Branch Psychological Sciences Division Office of Naval Research, Wash., D.C.

13 ABSTRACT

This is a report on the criterion development phase of a long-term research program concerned with computer personnel selection and evaluation. Two types of criterion measures are involved in this phase. Both are proficiency tests, one designed to test an individual's knowledge of the basic principles and techniques of programming, and the second, to test an individual's performance in depth in the systems analysis and systems design areas. The development of the first type, the Basic Programming Knowledge Test (BPKT), is described in this report. The second type of measure is now under construction and will be described in a subsequent report. (U)

The BPKT is intended to stand by itself as a criterion of programming proficiency. To achieve a close correspondence of test content to programming job requirements, subject-matter experts participated in the construction and review of the test questions. Test questions were selected that met the criteria of discrimination and appropriate difficulty, as indicated by the statistical analysis of results of a large preliminary testing. The final form of the test consists of 100 multiple-choice questions that are designed to be free of references to specific computers and languages now in use. Normative scores have been developed for Navy computer groups. The relationships of the BPKT test scores to a number of vocational and educational variables are described. Recommendations are made for the use of the BPKT in personnel selection and evaluation of experienced programmers and analysts, and as a criterion measure against which aptitude tests may be validated. (U)

DD FORM 1473 0101-807-6800

Unclassified
Security Classification

Unclassified
Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Basic Programming Knowledge Test Computer Personnel Selection Programming Test Programmer Selection Test Criterion Development Computer Programmer Proficiency Computer Programmer Characteristics Navy Programmers and Systems Analysts						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year, or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through _____."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through _____."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through _____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS)* *(S)* *(C)* or *(U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, roles, and weights is optional.

Unclassified
Security Classification